

# Игра "Арканоид"

Выполнили:

Шадрин Александр Иванович

Семенова Милана Александровна



Играть

Настройки

Справка

Выйти



# Описание игры

**"Арканоид"** - это классическая аркадная игра, где вашей задачей является управление платформой и отбивание мяча для разрушения блоков. Ваша цель - не дать мячу упасть за нижнюю границу экрана. Вам нужно отбить мяч с помощью платформы таким образом, чтобы он разбил все блоки. Игра **"Арканоид"** предлагает увлекательный геймплей, позволяет развлечься и проверить свою реакцию и навыки управления.



# Особенности игры



Игрок	Очки
Sasha	940
Mila	600

1

## Увлекательный геймплей

Игра предлагает простой и увлекательный геймплей, который легко понять и играть

2

## Разнообразие уровней

Разнообразные уровни с различными типами блоков, создающими разнообразие в игровом процессе

3

## Соревнование с игроками

Возможность соревноваться с другими игроками за высокие места в таблице игроков, где у каждого есть количество очков за разрушенные бананы



# Проблемы, решаемые игрой

1

Развитие логического мышления

Игра помогает развивать у игроков навыки логического мышления и принятия решений в сложных ситуациях.

2

Улучшение координации движений

Игра требует точности и скорости реакции, что помогает улучшить координацию движений у игроков.



```

class Platform(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        self.image = pygame.transform.scale(
            load_image("monkey.png"), (platform_width, platform_height)
        )
        self.image.set_colorkey(0)
        self.rect = self.image.get_rect()
        self.rect.x = screen_width // 2 - platform_width // 2 + 10
        self.rect.y = screen_height - platform_height - 10
        self.platform_speed = 10

    def update(self):
        d = pygame.key.get_pressed()
        if d[K_LEFT] and self.rect.left > 0:
            self.rect.left -= self.platform_speed
        if d[K_RIGHT] and self.rect.right < screen_width:
            self.rect.right += self.platform_speed

class Ball(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        self.image = pygame.transform.scale(
            load_image("coconut.png"), (ball_radius, ball_radius)
        )
        self.image.set_colorkey(0)
        self.rect = self.image.get_rect()
        self.rect.x = (screen_width - ball_radius) // 2
        self.rect.y = screen_height - platform_height - ball_radius * 2 - 10
        self.ball_speed_x = 6
        self.ball_speed_y = -6

    def update(self):
        self.rect.x += self.ball_speed_x
        self.rect.y += self.ball_speed_y

        if self.rect.left < 10 or self.rect.right >= screen_width:
            self.ball_speed_x *= -1
        if self.rect.top < 10:
            self.ball_speed_y *= -1
        if self.rect.colliderect(platform):
            self.ball_speed_x, self.ball_speed_y = detect_collision(
                self.ball_speed_x, self.ball_speed_y, self.rect, platform.rect
            )

class Block(pygame.sprite.Sprite):
    def __init__(self, x, y):
        super().__init__()
        self.image = pygame.transform.scale(
            load_image("banana.png"), (block_width, block_height)
        )
        self.image.set_colorkey(0)
        self.rect = self.image.get_rect()
        self.rect.x = x
        self.rect.y = y

```

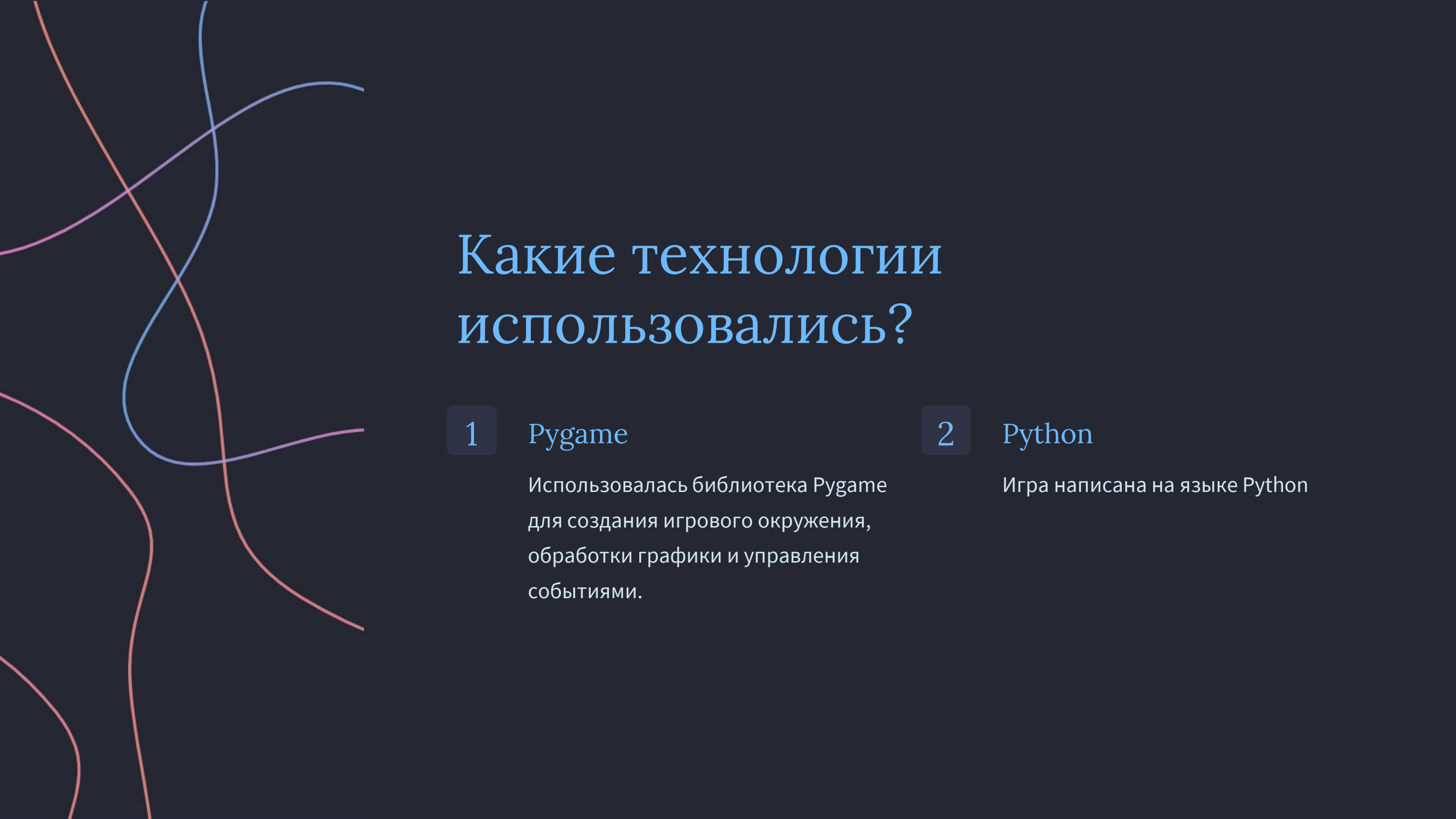
# Структура игры

## Классы

Реализация классов для платформы (Class Platform), мяча (Class Ball), блоков (Class Block), Не разрушаемых блоков (Class No\_Destructive\_Block) обеспечивает четкую и структурированную архитектуру проекта.

## Функции

Использование функций для окон игры: Справка (open\_guide\_window), настроек (open\_settings\_window), стартового окна (start\_screen) и финального окна (show\_result\_window) - также для обработки отскока (detect\_collision), отображения блоков (show\_blocks), обработки паузы (pause\_game) обеспечивает эффективное исполнение логики игры.



# Какие технологии использовались?

1

Pygame

Использовалась библиотека Pygame для создания игрового окружения, обработки графики и управления событиями.

2

Python

Игра написана на языке Python

# Основные элементы игры: платформа, мяч, блоки

1

## Платформа

Игрок управляет платформой, которая заменена спрайтом обезьянки, отбивая мяч и предотвращая его падение.



2

## Мяч

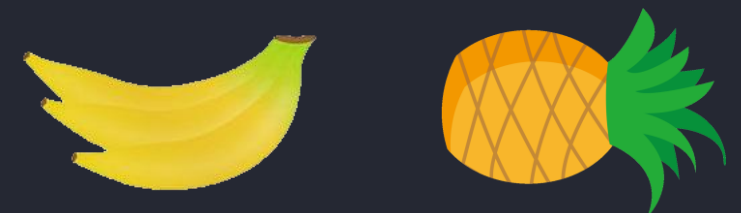
Мяч, представленный в виде кокоса или арбуза, используется для разрушения блоков и обеспечения прогресса в игре.



3

## Блоки

Цель игры - разрушить все блоки, которые отображаются в виде банана или ананаса, на уровне с помощью мяча и платформы.



# Реализация логики игры

```
def detect_collision(ball_speed_x, ball_speed_y, ball, rect):  
    if ball_speed_x > 0:  
        delta_x = ball.right - rect.left  
    else:  
        delta_x = rect.right - ball.left  
    if ball_speed_y > 0:  
        delta_y = ball.bottom - rect.top  
    else:  
        delta_y = rect.bottom - ball.top  
  
    if abs(delta_x - delta_y) < 10:  
        ball_speed_x, ball_speed_y = -ball_speed_x, -ball_speed_y  
    elif delta_x > delta_y:  
        ball_speed_y = -ball_speed_y  
    elif delta_y > delta_x:  
        ball_speed_x = -ball_speed_x  
    return ball_speed_x, ball_speed_y
```

## Отскок мяча

Реализована механика отскока мяча от платформы и блоков с помощью определения столкновения объектов и сменой вектора направления скорости



level\_3.txt –  
Файл Правка  
BBBBBBBBBB  
BBXBBBBXB  
BBBBXXBBBB  
BXBBBBBBXB

## Конструктор уровней

Реализована логика, с помощью которой читается уровень из txt-файла и выполняется расстановка блоков на экране



# Уровни сложности и прогрессия в игре

1

## Уровни сложности

Игра предлагает постепенное увеличение сложности с уровня к уровню, благодаря 3 уровням, обеспечивая более сложные раскладки блоков

2

## Прогрессия

Игрок развивается, проходя все новые уровни и сталкиваясь с разнообразными вызовами, обеспечивая увлекательный игровой опыт.

# Вывод и возможные улучшения игры

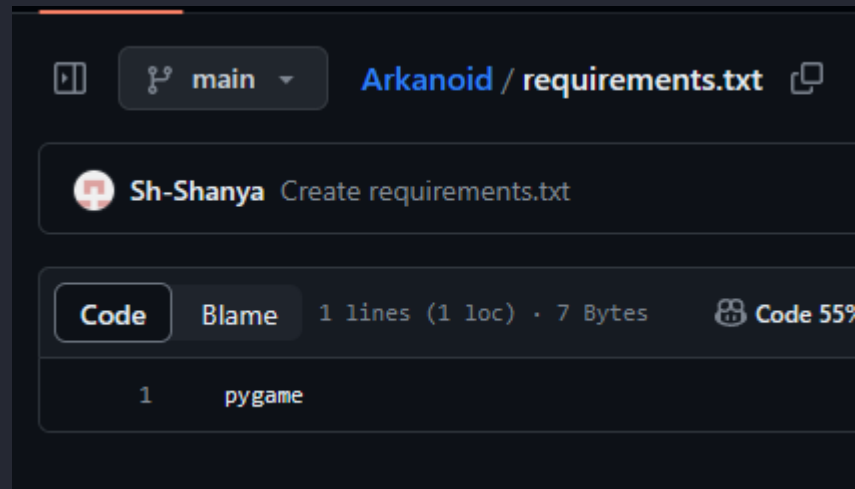
## 1 Увлекательный Игровой Процесс

Арканоид обеспечивает захватывающий игровой процесс, который может привлечь широкую аудиторию любителей аркадных игр.

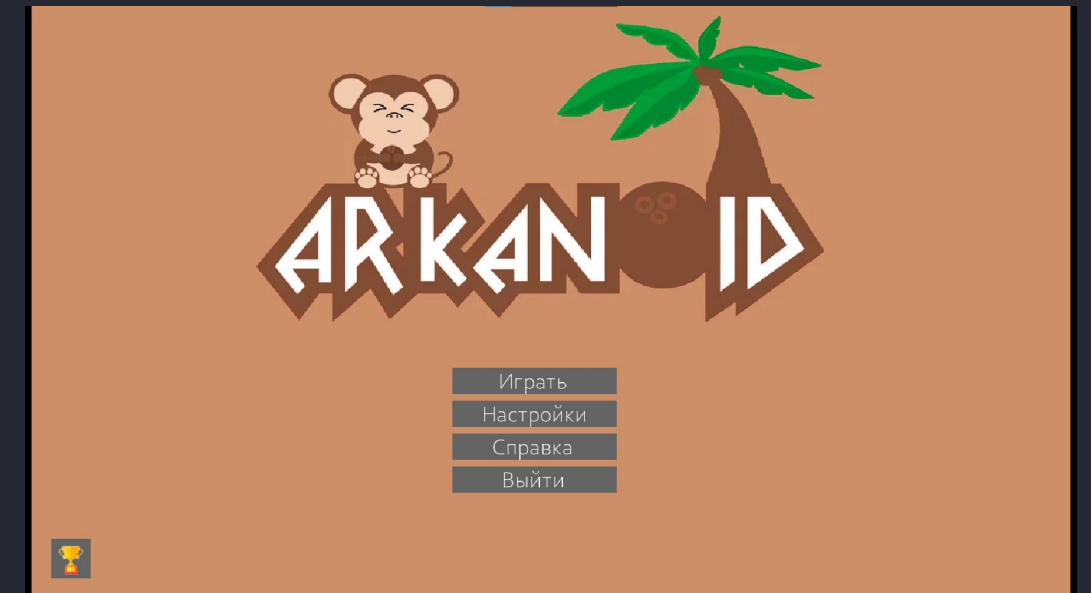
## 2 Потенциал для Расширения

Дополнительные уровни, бонусы, и новые механики могут значительно расширить геймплей и привнести новые возможности для игры.

# Критерии оценивания



requirements.txt

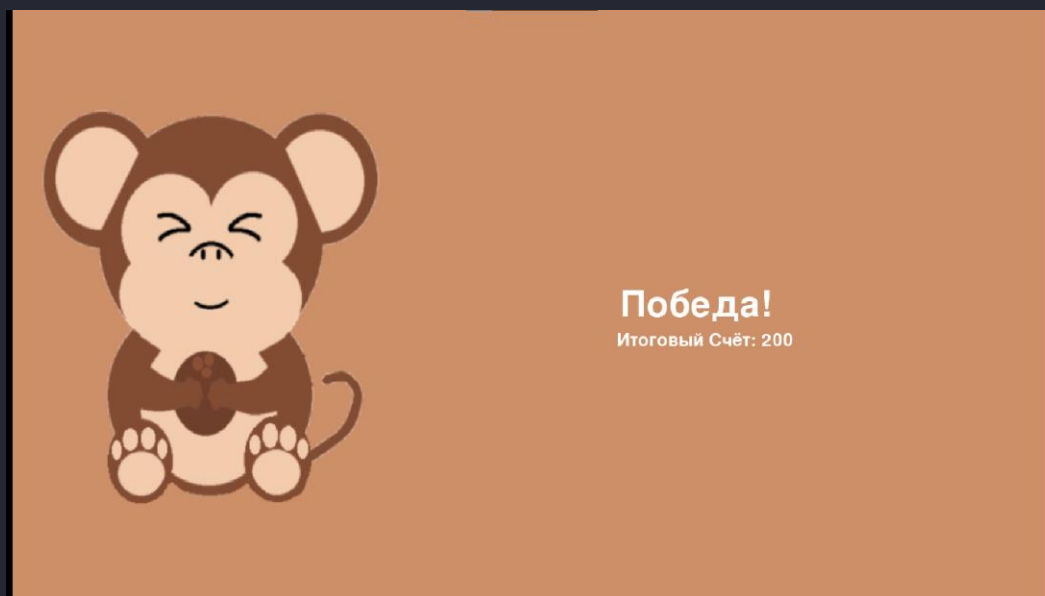


Стартовое окно

```
855  
856     level_data = read_level_from_file(f"data/level_{selected_level}.txt")  
857     all_score = sum([x.count("B") for x in level_data])  
858     start_game()  
859  
860  
861     pygame.quit()  
862
```

Объём кода

# Критерии оценивания



Финальное окно  
и  
Подсчет результатов



Спрайты



# Критерии оценивания

```
if self.rect.top < 10:  
    self.ball_speed_y *= -1  
  
if self.rect.colliderect(platform):  
    pygame.mixer.Sound("data/collision_1.wav").play()  
    self.ball_speed_x, self.ball_speed_y = detect_collision(  
        self.ball_speed_x, self.ball_speed_y, self.rect, platform.rect  
    )
```

collide



Анимация

# Критерии оценивания



Несколько уровней



data\_players.txt – Блокнот

Файл	Правка	Формат	Вид	Справка
Sasha	940	1		
Mila	410	2		

Хранение данных (txt)